dV/dt Accelerating the Rate of Progress towards Extreme Scale Collaborative Science

Miron Livny (UW)
Ewa Deelman, Gideon Juve, Rafael Ferreira da Silva (USC)
Ben Tovar, Casey Robinson, Douglas Thain (ND)
Frank Wuerthwein (UCSD)
Bill Allcock (ANL)

Funded by DOE

Thesis

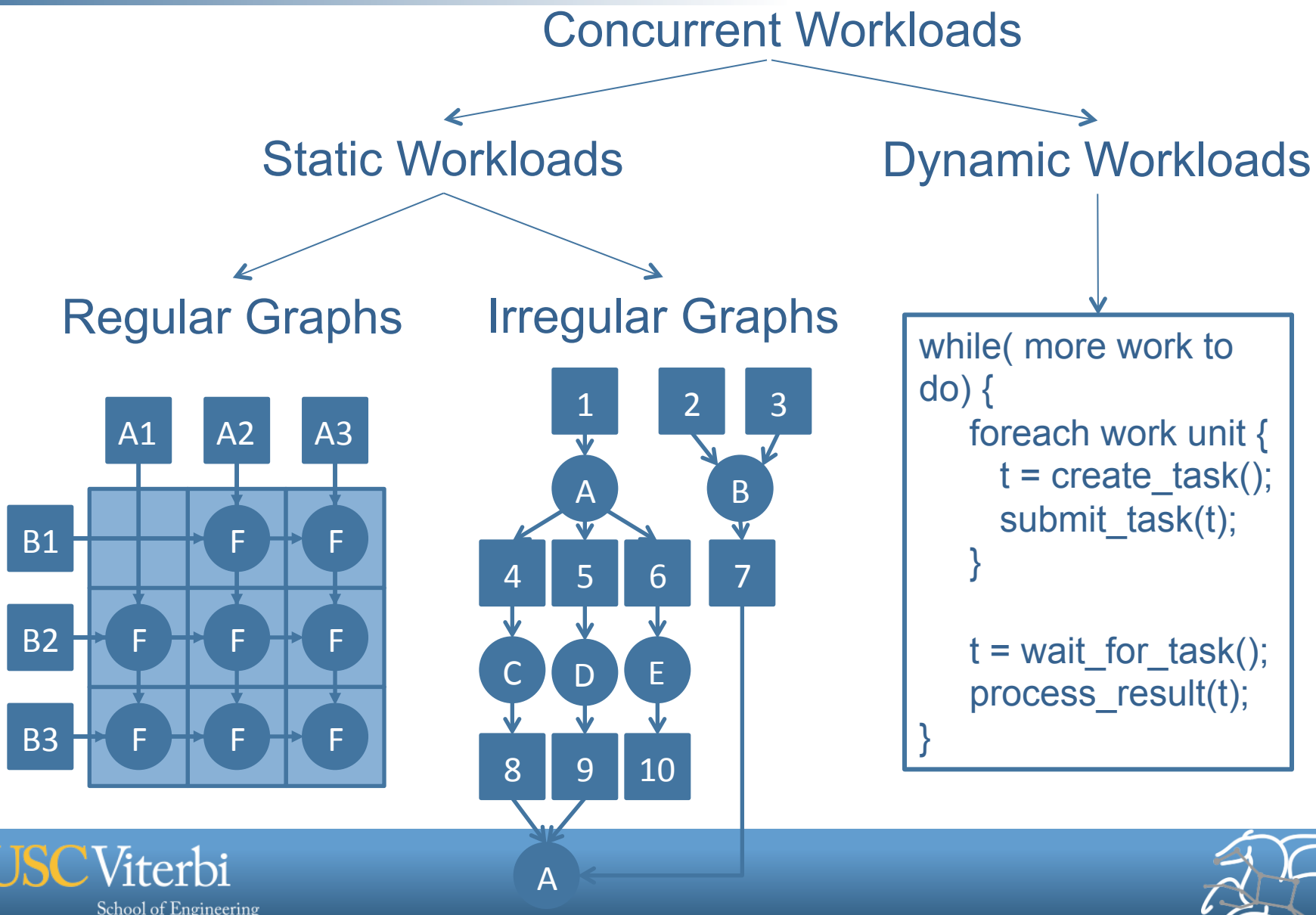
- Researchers band together into dynamic collaborations and employ a number of applications, software tools, data sources, and instruments
- They have access to a growing variety of processing, storage and networking resources
- Goal: "make it easier for scientists to conduct large-scale computational tasks that use the power of computing resources they do not own to process data they did not collect with applications they did not develop"

Challenges today

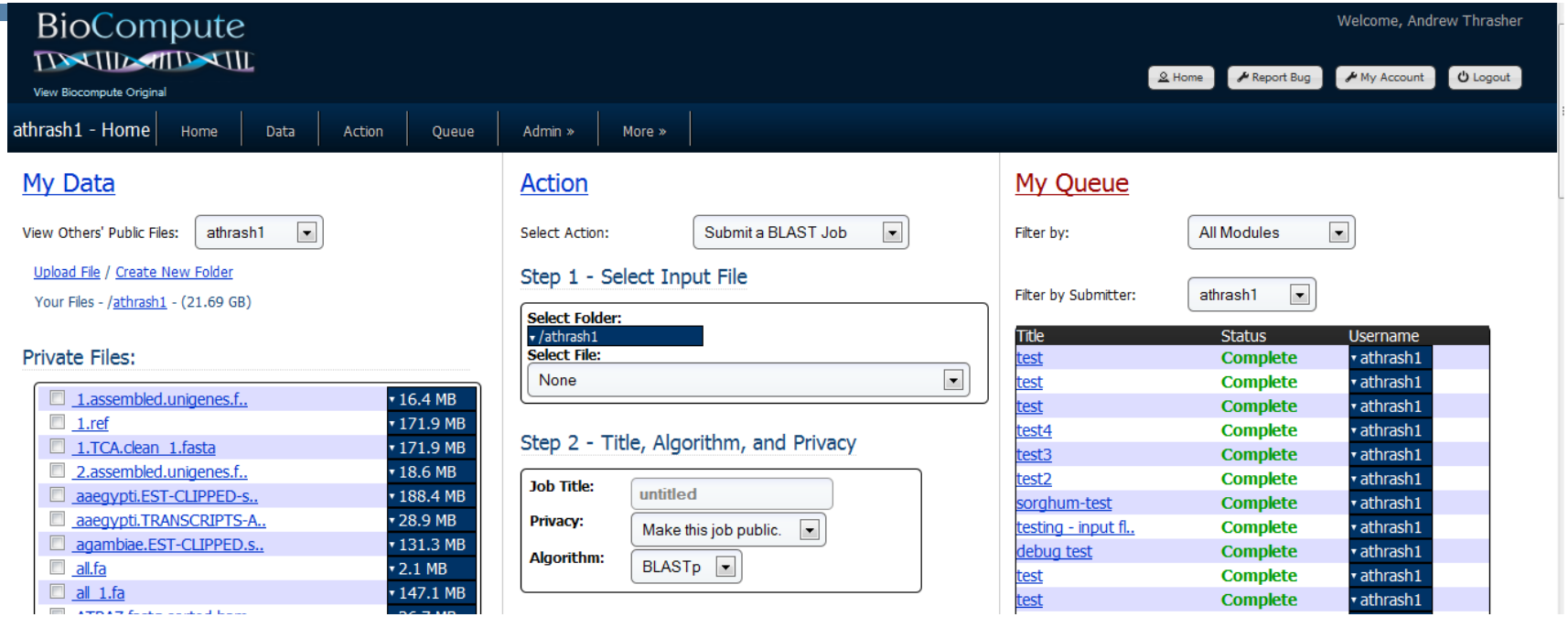
- Estimate the application resource needs
- Finding the appropriate computing resources
- Acquiring those resources
- Deploying the applications and data on the resources
- Managing applications and resources during run
- Make sure the application actually finishes successfully!

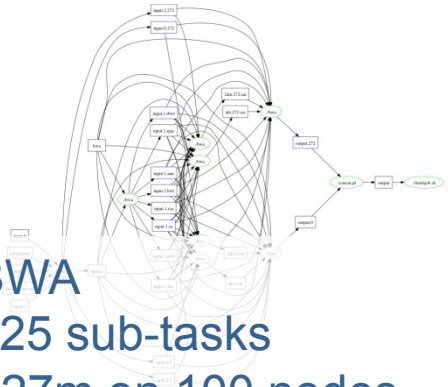
- **Approach:** Develop a framework that encompass the five phases of collaborative computing—estimate, find, acquire, deploy, and use

Application Characterization

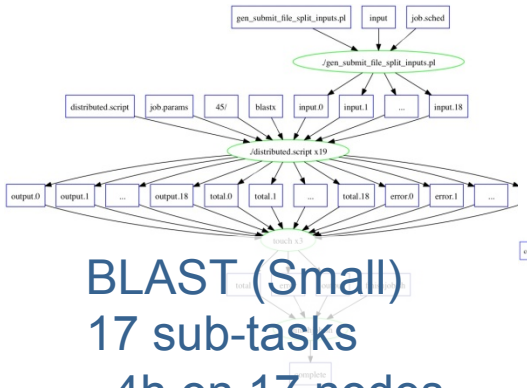


Portal Generated Workflows using Makeflow

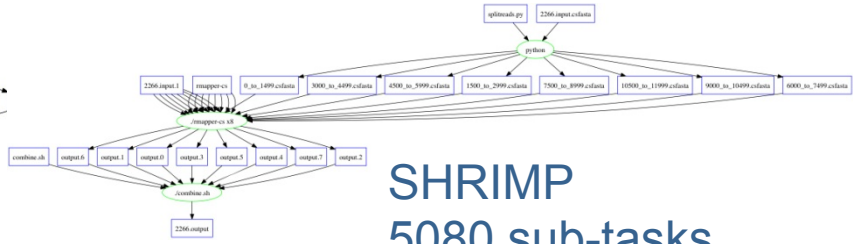




BWA
825 sub-tasks
~27m on 100 nodes



BLAST (Small)
17 sub-tasks
~4h on 17 nodes



SHRIMP
5080 sub-tasks
~3h on 200 nodes

USC Viterbi
School of Engineering

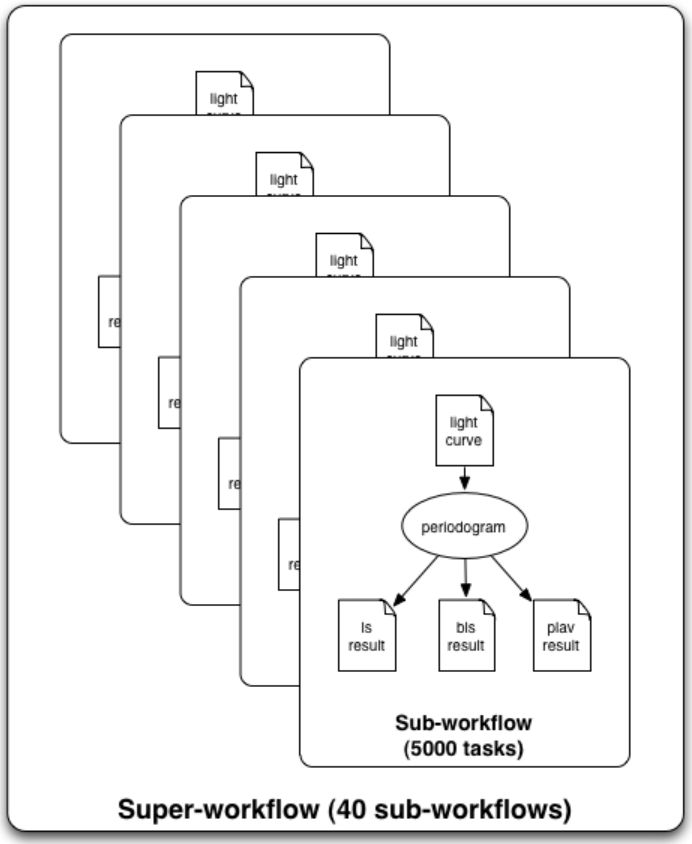
Periodograms: generate an atlas of extra-solar planets

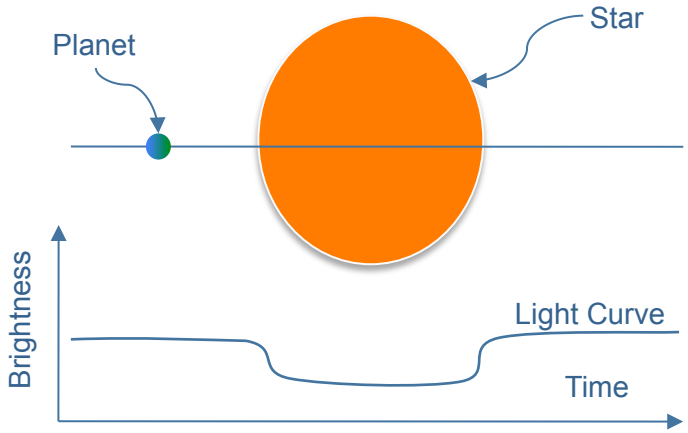
- **Find extra-solar planets by**
  - **Wobbles in radial velocity of star, or**
  - **Dips in star's intensity**

210k light-curves released in July 2010

Apply 3 algorithms to each curve

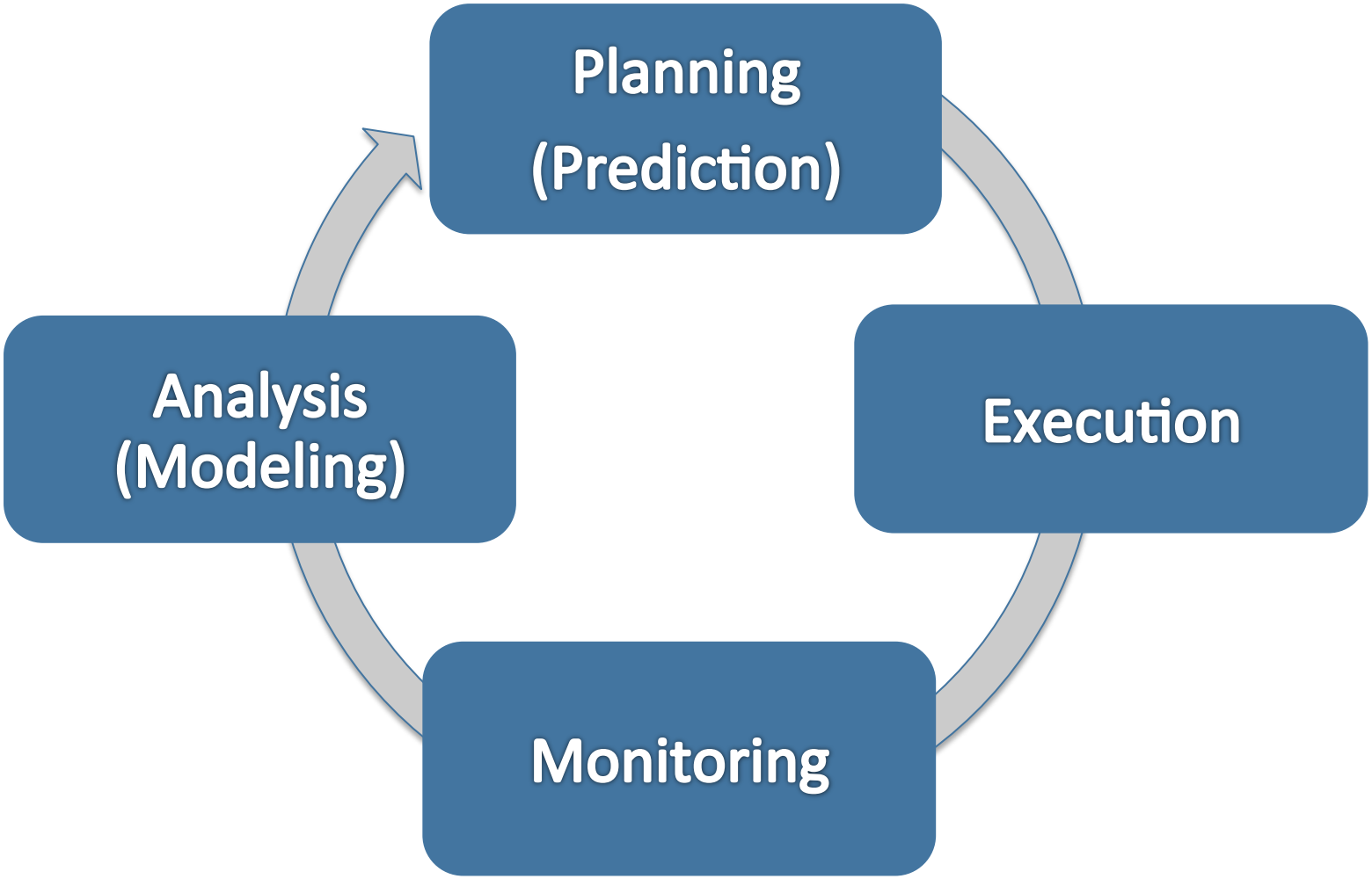
3 different parameter sets





- 210K input, 630K output files
- 1 super-workflow
- 40 sub-workflows
- ~5,000 tasks per sub-workflow
- 210K tasks total

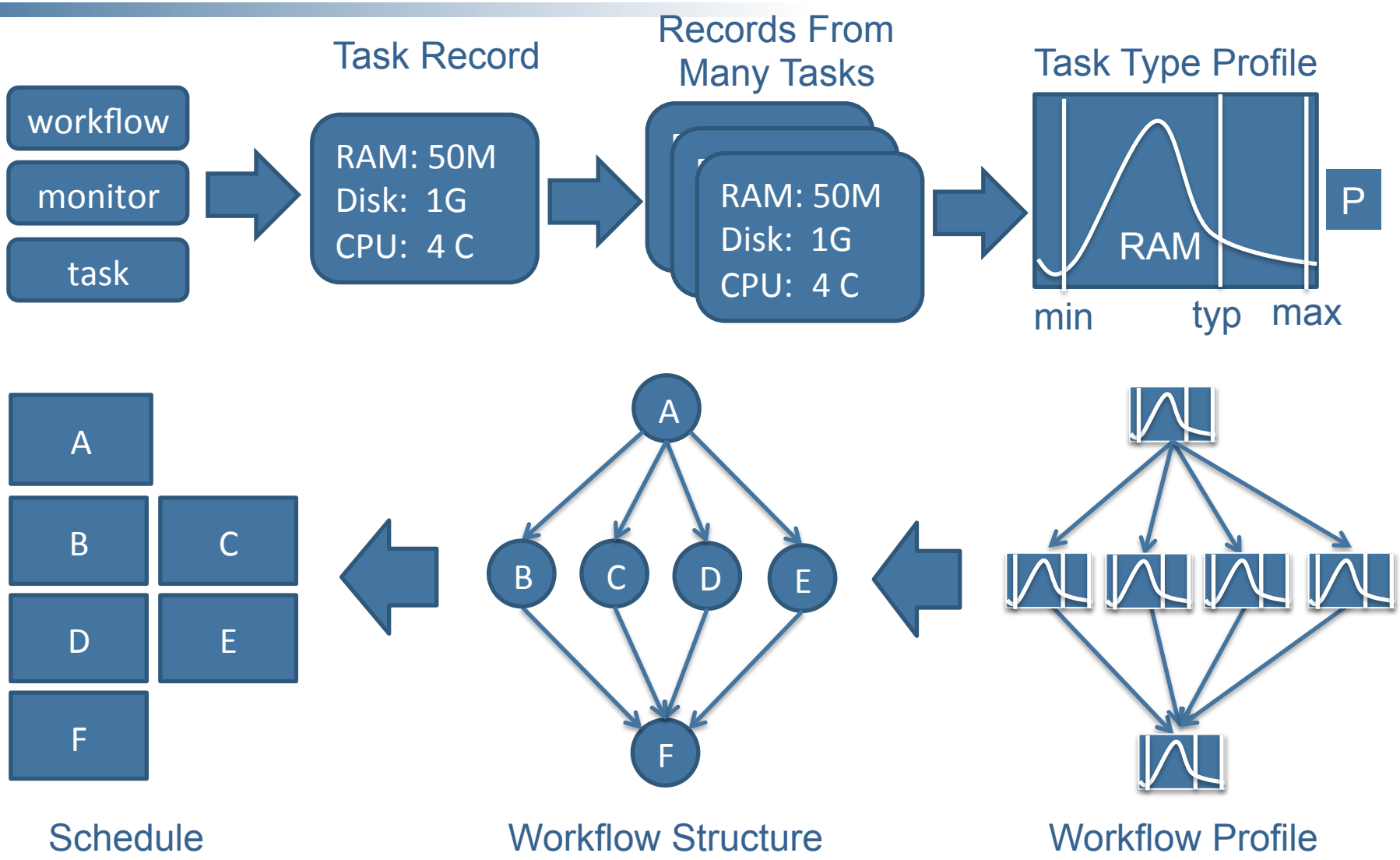
Characterizing Application Resource Needs



Task Characterization/Execution

- Understand the resource needs of a task
- Establish expected values and limits for task resource consumption
- Launch tasks on the correct resources
- Monitor task execution and resource consumption, interrupt tasks that reach limits
- Possibly re-launch task on different resources

Data Collection and Modeling



Resource Usage Monitoring

Resource Monitoring

- **Measure Resource Usage**
  - Runtime (wall time of process)
  - CPU usage (FLOPs, utime, stime)
  - Memory usage (peak resident set size, peak VM size)
  - I/O (data read/written, number of reads/writes)
  - Disk (size of files accessed/created)
- **Impose Limits**
  - Use models to predict usage
  - Use predictions to set limits
  - Detect violations of limits to prevent problems at runtime

Monitoring Accuracy with Synthetic Benchmarks

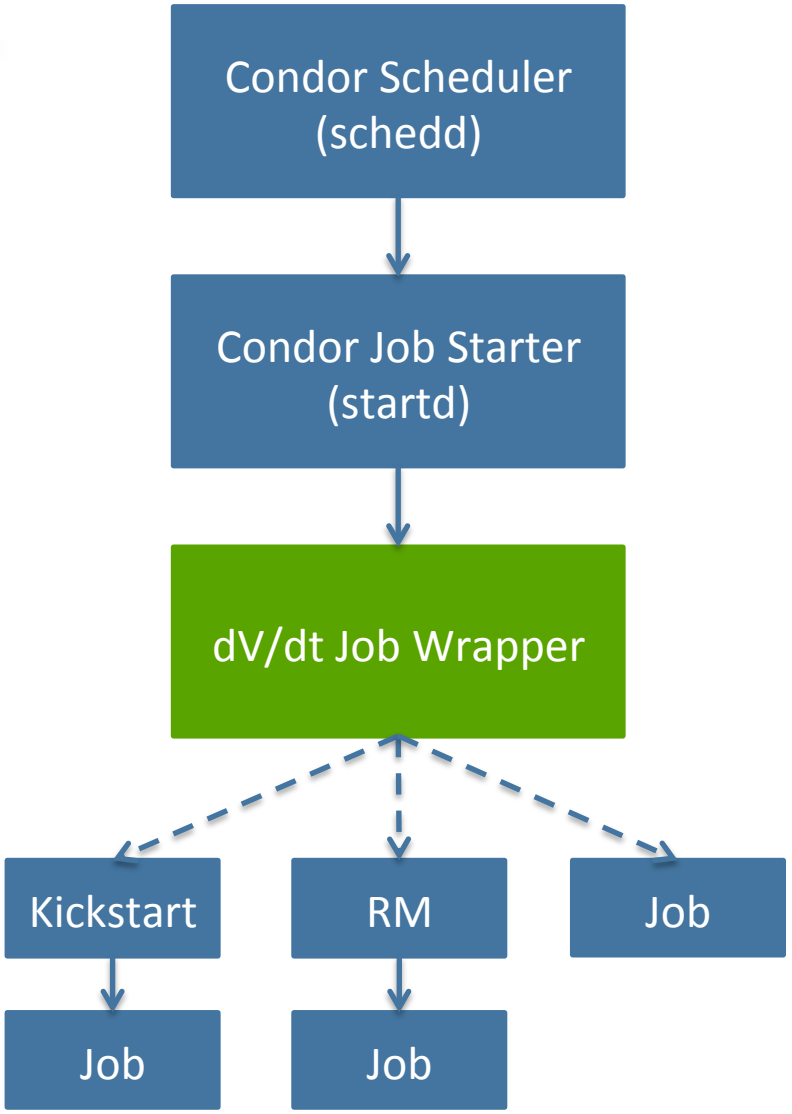
	Baseline	Polling (resource_monitor)		fork/exit LD_PRELOAD (resource_monitor)		fork/exit ptrace (kickstart)		syscall ptrace (kickstart)	
Instr.				**(a) CPU time**					
10^6	0.32 s	+0.04	(12.50%)	+0.02	(4.91%)	0.00	(0.00%)	0.00	(0.00%)
10^7	2.93 s	+0.06	(2.12%)	+0.04	(1.20%)	0.00	(0.00%)	+0.01	(0.14%)
10^8	28.20 s	+0.17	(0.60%)	+0.09	(0.31%)	+0.03	(0.10%)	+0.04	(0.14%)
10^9	279.53 s	+1.29	(0.46%)	+1.32	(0.47%)	+0.20	(0.07%)	+0.41	(0.15%)
Memory				**(b) Memory: resident size**					
1GB	1GB		−13.96%		+0.08%		+0.03%		+0.03%
2GB	2GB		−17.63%		+0.03%		+0.02%		+0.02%
4GB	4GB		−2.25%		+0.02%		0.00%		0.00%
8GB	8GB		−1.89%		+0.01%		0.00%		0.00%
16GB	16GB		−1.99%		+0.01%		0.00%		0.00%
File size				**(c) I/O: bytes read, 4KB buffer**					
1MB	1MB		−13.64%		0.00%		0.00%		0.00%
100MB	100MB		−9.07%		0.00%		0.00%		0.00%
1GB	1GB		−5.84%		0.00%		0.00%		0.00%
10GB	10GB		−2.13%		0.00%		0.00%		0.00%
Buffer size				**(d) I/O: bytes read, 1GB file**					
4KB	1GB		−5.84%		0.00%		0.00%		0.00%
8KB	1GB		−0.82%		0.00%		0.00%		0.00%
16KB	1GB		−15.41%		0.00%		0.00%		0.00%
32KB	1GB		−18.41%		0.00%		0.00%		0.00%

Monitoring Overhead

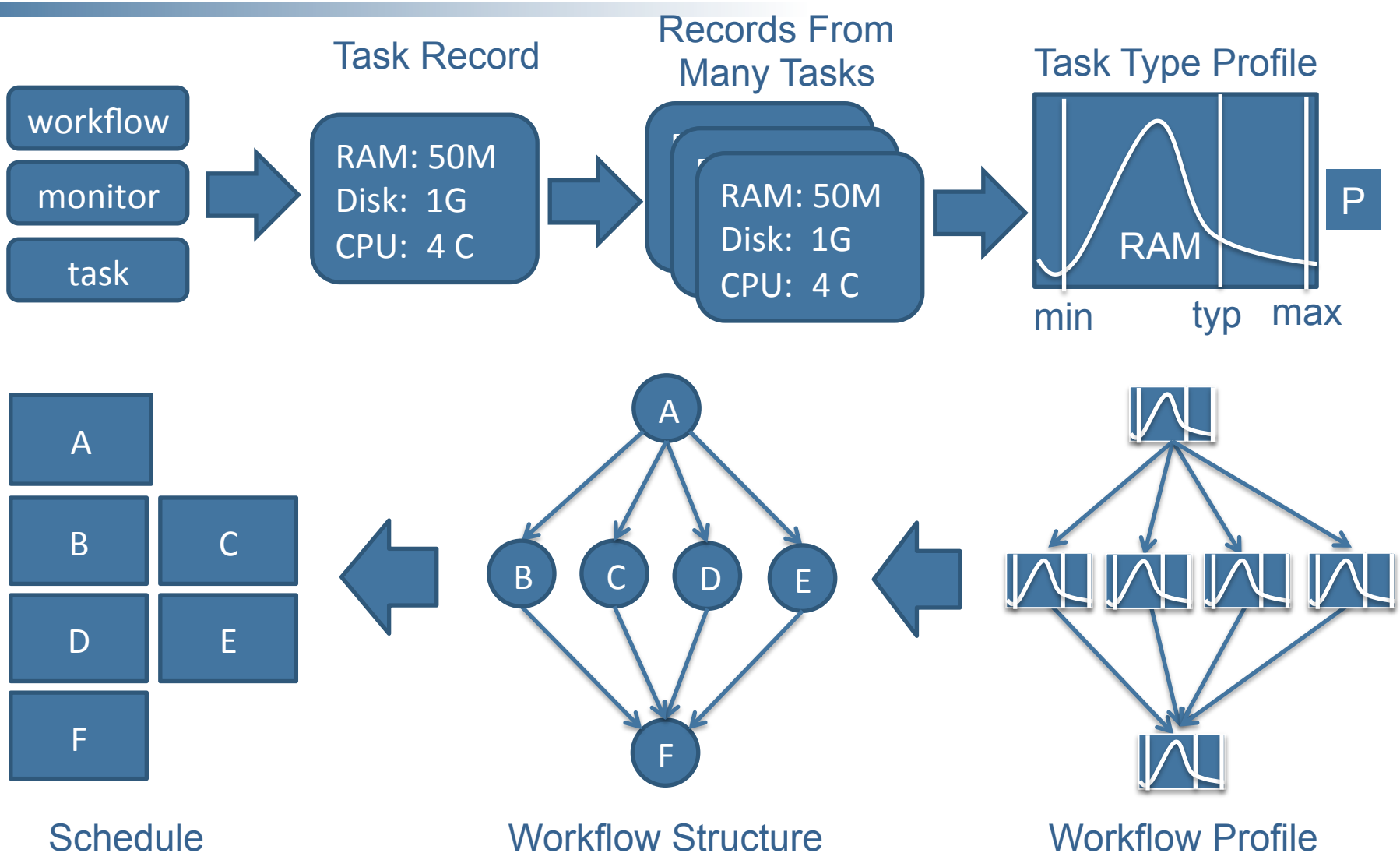
	Baseline	Polling (resource_monitor)		fork/exit LD_PRELOAD (resource_monitor)		fork/exit ptrace (kickstart)		syscall ptrace (kickstart)	
Instr.		**(a) CPU overhead**							
10^6	0.32 s	+0.22	(68.75%)	+0.25	(78.13%)	+0.18	(56.25%)	+0.13	(40.63%)
10^7	2.93 s	+0.28	(9.56%)	+2.42	(82.59%)	+0.14	(4.78%)	+0.14	(4.78%)
10^8	28.20 s	+0.17	(0.60%)	+0.22	(0.78%)	+0.10	(0.35%)	+0.12	(0.43%)
10^9	279.53 s	+0.28	(0.10%)	+0.78	(0.28%)	+0.07	(0.03%)	+0.61	(0.22%)
Resident size		**(b) Memory overhead**							
1GB	3.57 s	+0.17	(4.76%)	+0.26	(7.28%)	+0.06	(1.68%)	+0.07	(1.96%)
2GB	6.19 s	+0.10	(1.62%)	+0.14	(2.26%)	+0.09	(1.45%)	+0.06	(0.97%)
4GB	12.64 s	+0.50	(3.96%)	+0.86	(6.80%)	+0.24	(1.90%)	+0.43	(3.40%)
8GB	25.06 s	+0.51	(2.04%)	+1.88	(7.50%)	+0.87	(3.47%)	+0.96	(3.83%)
16GB	52.81 s	+1.11	(2.10%)	+4.69	(8.88%)	+1.38	(2.61%)	+2.25	(4.26%)
File size		**(c) I/O overhead, 4KB buffer**							
1MB	0.01 s	+0.17	(1700%)	+0.24	(2400.00%)	+0.13	(1300.00%)	+0.14	(1400.00%)
100MB	1.53 s	+0.09	(5.88%)	+0.10	(6.54%)	+0.09	(5.88%)	+1.82	(118.95%)
1GB	16.02 s	+0.04	(0.25%)	+0.38	(2.37%)	+0.36	(2.25%)	+15.98	(99.75%)
10GB	153.98 s	+0.54	(0.35%)	+0.64	(0.42%)	+0.58	(0.38%)	+143.95	(93.49%)
Buffer size		**(d) I/O overhead, 1GB file**							
4KB	16.02 s	+0.04	(0.25%)	+0.38	(2.37%)	+0.36	(2.25%)	+15.98	(99.75%)
8KB	9.14 s	+0.20	(2.19%)	+0.38	(4.16%)	+0.24	(2.63%)	+8.72	(95.40%)
16KB	6.40 s	+0.23	(3.59%)	+0.34	(5.31%)	+0.30	(4.69%)	+4.13	(64.53%)
32KB	4.37 s	+0.18	(4.12%)	+0.43	(9.84%)	+0.60	(13.73%)	+2.11	(48.28%)

Condor Job Wrapper

- **Selectively wraps Condor jobs with monitoring tools**
  - **Uses USER_JOB_WRAPPER functionality of Condor**
  - **Does not wrap jobs that have failed**
  - **Selectively monitors based on user, executable, etc.**
  - **Selectively monitors a given percentage of jobs (e.g. 50% of jobs)**
  - **Detects monitor errors and restarts job without wrapper**
- **Allows us to easily deploy monitoring tools on production Condor pools**

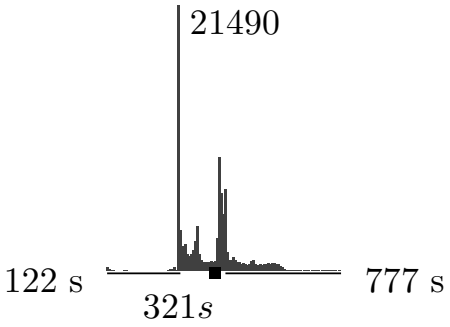
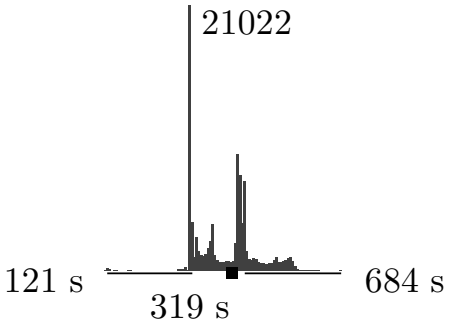
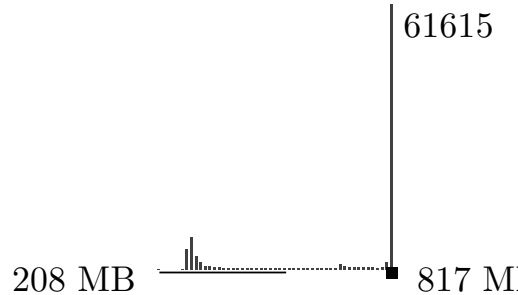


Data Collection and Modeling



Resource Monitoring Archive

- **Stores monitoring records**
- **Provides a query interface for analyzing data**

resource	wall time	cpu time	resident memory
histogram			
mean	410.55 s	406.17 s	682.62 MB
std. dev.	79.16	73.86	208.83
skewness	0.42	0.17	-1.11
kurtosis	0.26	-0.10	10.96

Resource Usage Limits

```
memory:    4000000
num_proc: 300
cpu_time:  40000000
disk:      100000000000
```

global: limits file

```
#: num_proc: 250, fs_nodes: 1
sqrts: msqrt
    ./msqrt 200
```

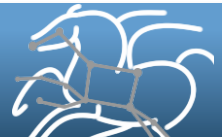
local: per task rule

Limits specification

```
command: ./msqrt 200
start:    1361995712680901 Wed Feb 27 15:08:32 2013
end:      1361995725794759 Wed Feb 27 15:08:45 2013
exit-type:      signal 2 Termination
exit-status:    1
monitor-watch-end: cpu_time 42740000 > 40000000

                max
processes:      201
cpu_time:       42740000
memory:         410295
io-chars:       67467528
vnodes:         22
bytes:          94187
fs_nodes:       0
```

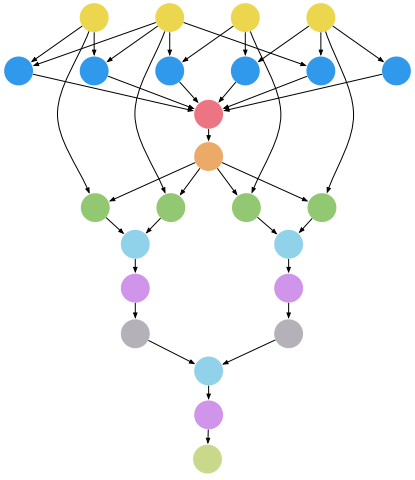
Record with alarm



Resource Usage Modeling

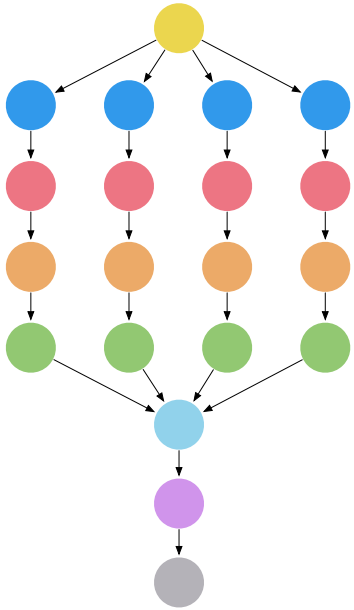
Workflow Execution Profiling

- Workflows were executed using Pegasus WMS and profiled
  - Monitors and records fine-grained data
  - E.g. process I/O, runtime, memory usage, CPU utilization
- **3** runs of each workflow with different datasets



Small (20 node) Montage Workflow

Periodogram Workflow



Epigenomics Workflow

Execution Profile: Montage Workflow

Task estimation could be based on mean values

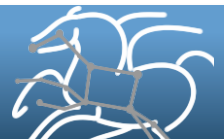
Task	Count	Runtime		I/O Read		I/O Write		Memory Peak	
		Mean (s)	Std. Dev.	Mean (MB)	Std. Dev.	Mean (MB)	Std. Dev.	Mean (MB)	Std. Dev.
mProjectPP	7965	2.59	0.69	4.24	0.19	16.20	0.80	9.96	0.40
mDiffFit	23733	1.25	0.92	24.08	5.76	1.35	1.11	5.32	0.90
mConcatFit	3	122.04	5.27	2.70	0.01	3.15	0.01	7.26	0.01
mBgModel	3	2008.08	88.50	4.14	0.04	0.27	0.00	14.41	0.01
mBackground	7965	2.14	1.68	13.67	6.78	13.05	6.44	11.75	5.78
mImgtbl	51	4.65	2.04	22.64	4.61	0.25	0.05	6.37	0.13
mAdd	51	47.69	14.03	2191.76	560.39	1574.22	383.86	21.66	3.40
mShrink	48	11.53	2.25	835.57	0.31	1.00	0.00	3.05	0.01
mJPEG	3	1.03	0.07	46.18	0.02	0.78	0.00	2.66	0.01

uses Kickstart profiling tool

Task estimation based on average may lead to significant estimation errors

16-core cluster

- 5 Dual core MP Opteron™ Processor 250 2.4GHz / 8GB RAM
- 3 Dual core MD AMD Opteron™ Processor 275 2.2 GHz / 8GB RAM



Automatic Workflow Characterization

- Characterize tasks based on their estimation capability
  - **Runtime, I/O write, memory peak ➔ estimated from I/O read**

- Use correlation statistics to identify statistical relationships between parameters
  - High correlation values yield accurate estimations, **Estimation based on the ratio: parameter/input data size**

Task	Runtime		I/O Write		Memory Peak	
	ρ	σ	ρ	σ	ρ	σ
fastqSplit	0.98	9.00	1.00	297.15	0.00	0.01
filterContams	-0.03	0.27	0.99	1.46	0.00	0.01
sol2sanger	0.21	0.41	0.90	1.49	0.00	0.01
fast2bfq	0.18	0.27	0.56	0.87	0.00	0.01
map	0.02	18.96	0.06	0.70	0.01	1.43
mapMerge	0.98	13.33	0.99	189.81	-0.36	2.15
pileup	0.99	4.73	0.17	249.78	0.87	25.70

Constant values

Correlated if $\rho > 0.8$

Epigenomics workflow

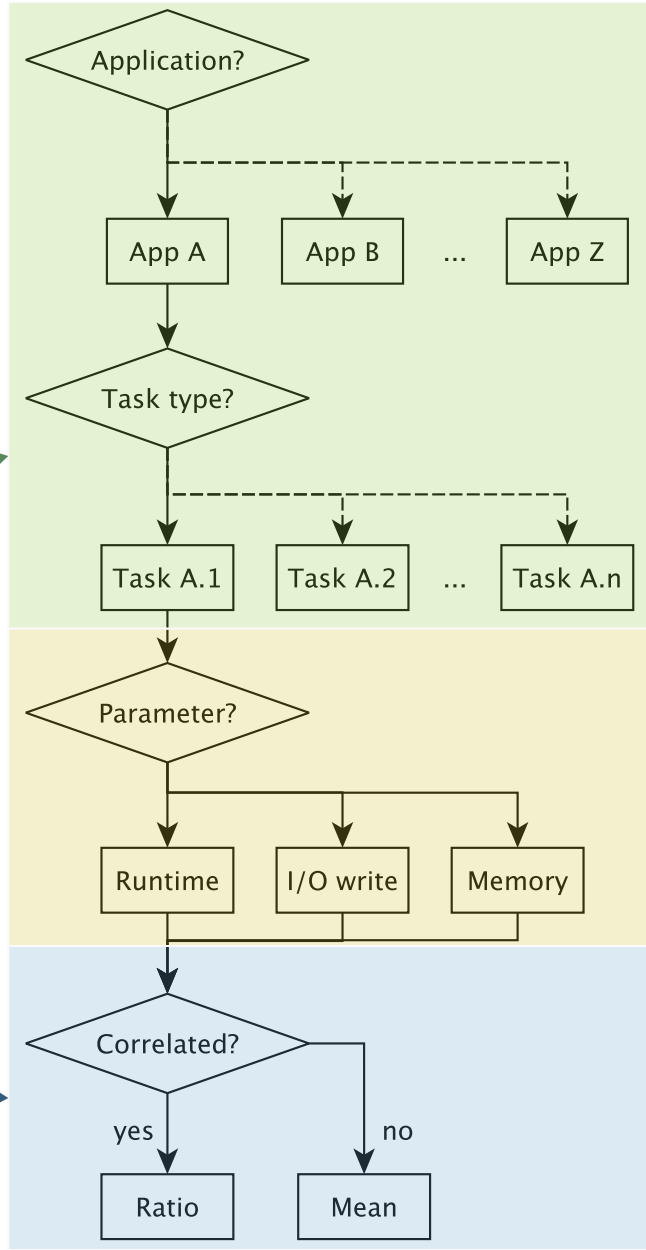
Task Estimation Process

- Based on Regression Trees
  - Built offline from historical data analyses

Tasks are classified by application, then task type

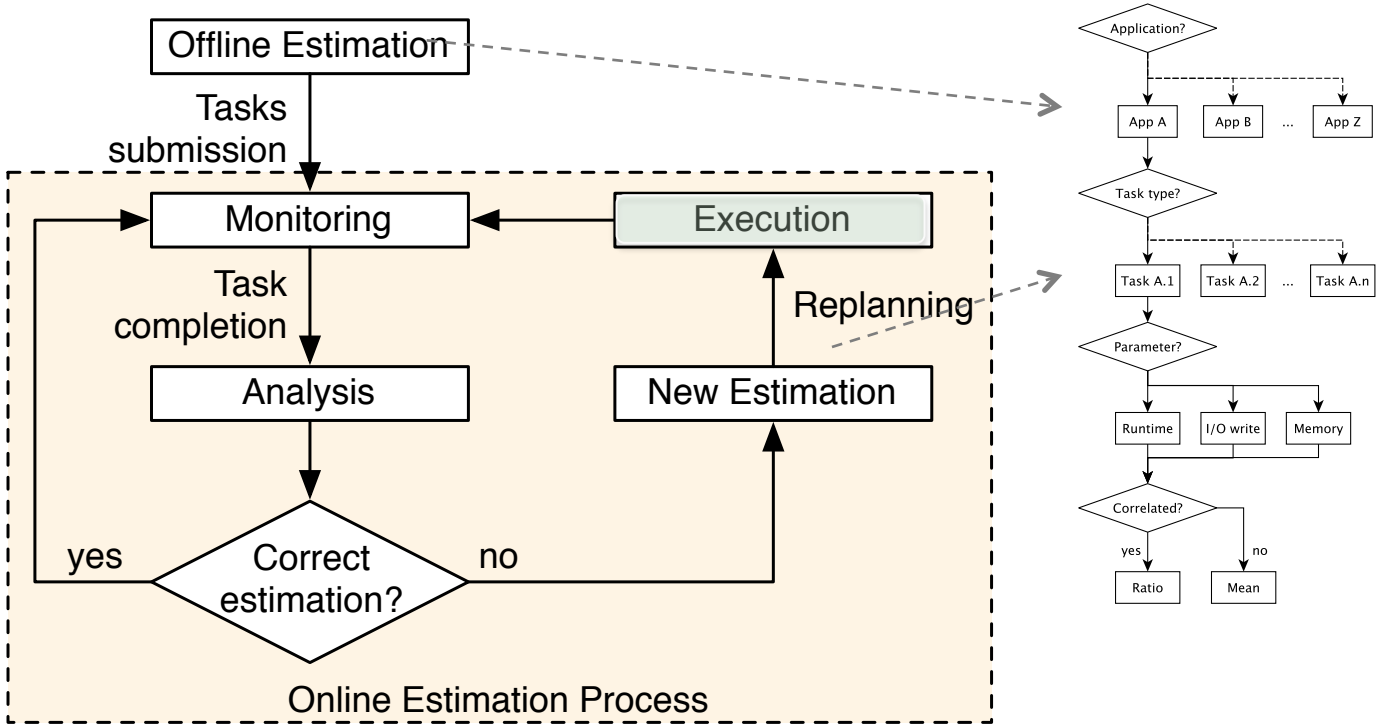
Estimation of runtime, I/O write, or memory peak

If strongly correlated to the input data:
- Estimation based on the ratio parameter/input data size
- Otherwise, estimation based on the mean



Online Estimation Process

- Based on the MAPE-K loop
  - Task executions are constantly monitored
  - Estimated values are updated, and a new prediction is done



Experiment: Use Estimations Online, while the workflow is executing

- Trace analysis of 3 workflow applications
  - Montage
  - Epigenomics
  - Periodogram
- Leave-one-out cross-validation
  - Evaluate the accuracy of our online estimation process
  - 3 different workflow execution traces for each workflow
- Simulator
  - Replays workflow executions

Results: Average Estimation Errors - Montage

Task	Estimation	Runtime Avg. Error (%)	I/O Write Avg.Error (%)	Memory Avg.Error (%)
mProjectPP	Offline	18.95	1.63	2.80
	Online	18.95	1.63	2.80
mDiffFit	Offline	191.02	159.46	91.07
	Online	46.52	69.14	73.72
mConcatFit	Offline	4.38	0.00	7.62
	Online	4.03	0.00	6.22
mBgModel	Offline	23.83	0.00	22.08
	Online	1.17	0.00	3.43
mBackground	Offline	65.13	102.80	104.62
	Online	44.90	1.23	1.84
mImgtbl	Offline	61.27	127.29	126.58
	Online	29.15	5.53	8.35
mAdd	Offline	9.67	113.14	110.20
	Online	9.31	3.43	9.06
mShrink	Offline	13.72	0.34	0.00
	Online	7.61	0.33	0.00
mJPEG	Offline	1.61	0.00	19.09
	Online	1.37	0.00	11.40

Poor output data estimations leads to a chain of estimation errors in scientific workflows

Offline Process
Avg. Runtime Error: 43%
Avg. I/O Write Error: 56%
Avg. Memory Error: 53%

Online Process
Avg. Runtime Error: 18%
Avg. I/O Write Error: 9%
Avg. Memory Error: 13%

- Online strategy counterbalances the propagation of estimation errors

Conclusions

A planning framework that:

- Starts with an unknown application
- Characterizes it, models it, and manages execution dynamically

Future:

- Experiments at scale on Condor pool at UW and OSG resources (model heterogeneous resources)
- Integrate resource provisioning into planning
- Experiment with predictions and resource provisioning
- https://sites.google.com/site/acceleratingexascale/